universidade anhembi morumbi
Mundialmente criativa e inovadora
Laureate International Universities®

Introdução a Programação para Games

Prof. Alexandre Nascimento Tomoyose

2º semestre 2016

universidade anhembi morumbi
Mundialmente criativa e inovadora
Laureate International Universities®

Funções

Prof. Alexandre Nascimento Tomoyose

2º semestre 2016

Funções

- **Definição**
  - Recurso da programação que permite isolar partes do código e facilita a repetição
  - Algumas funções são predefinidas pela linguagem, exemplos: print(), random()

Funções

- **Definição**
  - A sintaxe completa para criação de uma função segue o padrão:
    - tipo nome(parâmetros) { comandos }
  - O tipo determina o tipo de dado do retorno da função, caso não tenha retorno o tipo será void
  - O nome representa o nome da função que é um identificador

Funções

- **Definição**
  - Os **parâmetros** são variáveis que precisam ter seus valores estabelecidos para a execução da função, pode não ter nenhum
  - Os **comandos** representam o trecho de código a ser executado quando a função for chamada

Funções

- **Definição**
  - **ATENÇÃO, variáveis criadas em uma função só existem dentro desta função**
  - **Para uma variável existir em mais de uma função é necessário que ela seja uma variável GLOBAL, ou seja, que foi declarada fora das funções**

Funções

- **Utilização**
  - **Para criar uma função o primeiro passo é definir a assinatura da função, na qual inclui seu nome**
  - **Exemplo:**

```
void funcao()
{
    println("Executou");
}
```

Funções

- **Utilização**
  - O passo seguinte é definir os comandos que serão executados quando a função é chamada, definindo o seu **corpo**
  - Exemplo:

```
void funcao()
{
    println("Executou");
}
```

Funções

- **Utilização**
  - **Por fim, para executar a função basta usar seu nome seguido de parênteses, o que denominamos de chamar a função**
  - **Exemplo:**

```
funcao();

void funcao()
{
    println("Executou");
}
```

Funções

- **Parâmetros**
  - Opcionalmente, uma função pode exigir que um ou mais valores sejam definidos ao ser chamada, estes denominamos **parâmetros** ou **argumentos**
  - Os parâmetros são separados por vírgula (,) e todo parâmetro tem um tipo de dado definido

Funções

- **Parâmetros**
  - **Precisam ser relacionados na declaração da função**
  - **Exemplo:**

```
void funcao(String nome, int idade)
{
    println(nome + “, ” + idade + “ anos”);
}
```

Funções

- **Parâmetros**
  - **Os valores precisam ser definidos na chamada da função**
  - **Exemplo:**

```
funcao(“Garfield”, 13);
funcao(“Odie”, 8);

void funcao(String nome, int idade)
{
    println(nome + “, ” + idade + “ anos”);
}
```

Funções

- **Exercício**
  - Crie uma função chamada potencia()
  - Recebe dois parâmetros do tipo int: base e expoente
  - Calcula e imprime o resultado
  - Teste com expoentes positivos:
    - potencia(5, 2);
    - potencia(3, 4);
    - potencia(8, 0)

Funções

- **Exercício**
  - **Teste com expoentes negativos:**
    - **potencia(10, -1);**
    - **potencia(5, -2);**
    - **potencia(3, -4)**

Funções

- **Retorno**
  - Opcionalmente, uma função pode devolver um valor para o ponto do código onde foi chamada, este é o **retorno da função**
  - O tipo de dado do retorno é estabelecido na assinatura da função antes do nome
  - O tipo **void** é usado caso a função não tenha um valor de retorno

Funções

- **Retorno**
  - **Precisa ser estabelecido na declaração da função**
  - **Exemplo:**

```
int funcao(int a, int b)
{
    return a + b;
}
```

Funções

- **Retorno**
  - O comando **return** precisa ser usado no corpo da função para indicar qual o valor do retorno
  - Exemplo:

```
int funcao(int a, int b)
{
    return a + b;
}
```

Funções

- **Retorno**
  - **O valor do retorno pode ser atribuído a uma variável, ou usado em uma expressão como qualquer outro valor**
  - **Exemplo:**

```
int total = funcao(1, 2) * funcao(3, 4);

int funcao(int a, int b)
{
    return a + b;
}
```

Funções

- **Exercício**
  - Modifique a função potencia() para retornar o resultado ao invés de imprimir
  - Calcule o valor das seguintes expressões:
    - potencia(potencia(5, 2), 2);
    - potencia(potencia(5, -2), 2);
    - potencia(potencia(5, 2), -2);
    - potencia(2, 5) + potencia(3, 4);
    - potencia(2, -5) + potencia(3, -4);

universidade anhembi morumbi
Mundialmente criativa e inovadora
Laureate International Universities®

Colisão

Prof. Alexandre Nascimento Tomoyose

2º semestre 2016

Colisão

- **Definição**
  - **Contato entre corpos**
  - **Exemplos?**

Colisão

- **Definição**
  - **Em programação, o processo de uma colisão é dividido em duas partes:**
    - **Detecção da colisão**
    - **Tratamento da colisão**

Colisão

- **Detecção da colisão**
  - **Identificação de contato entre corpos**
  - **Em programação, sempre realizamos a detecção em pares, ou seja, de dois em dois corpos por vez**

Colisão

- **Tratamento da colisão**
  - **Consequências e implicações resultantes da colisão**
  - **A colisão entre sólidos, ou seja, impedir que um corpo atravesse o outro é apenas uma das consequências possíveis**

Colisão

- **Processo**
  - Normalmente, a detecção de colisão não é feita utilizando o contorno exato de um Sprite, pois torna o processo muito complexo e lento
  - Simplificações da colisão são criadas usando retângulos ou círculos (bounding box ou bounding circle)

Colisão

- **Exercício**
  - **Faça um programa que desenha dois retângulos com posições e tamanhos aleatórios**
  - **Detecte se há colisão entre os retângulos, se houver faça o preenchimento de vermelho, caso contrário de azul**

Colisão

- **Exercício**
  - **Faça um programa que desenha dois círculos com posições e tamanhos aleatórios**
  - **Detecte se há colisão entre os círculos, se houver faça o preenchimento de vermelho, caso contrário de azul**

Colisão

- **Mouse**
  - No processing existem duas variáveis globais que armazenam a posição atual do ponteiro do mouse: **mouseX** e **mouseY**
  - Além disso, uma terceira variável chamada **mousePressed** indica se o botão do mouse está pressionado

Colisão

- **Exercício**
  - Faça um programa que sempre desenha um círculo na posição do mouse
  - Pinte o círculo de verde caso o botão esteja pressionado e vermelho caso contrário
  - Faça o círculo perseguir o mouse, demorando alguns frames para alcança-lo

Colisão

- **Exercício**
  - **Faça um programa que desenha um retângulo com posição e tamanho aleatórios**
  - **Detecte se o ponteiro está sobre o retângulo, se estiver mude o retângulo para uma cor diferente**

Colisão

- **Exercício**
  - Faça um programa que desenha um círculo com posição e tamanho aleatórios
  - Detecte se o ponteiro está sobre o círculo, se estiver mude o círculo para uma cor diferente